Dynamically balanced and plausible trajectory planning for human-like characters

Chonhyon Park, Jae Sung Park, Steve Tonneau, Nicolas Mansard, Franck Multon, Julien Pettré, Dinesh Manocha

Dynamically Balanced and Plausible Trajectory Planning for Human-Like Characters

Chonhyon Park*, Jae Sung Park*, Steve Tonneau†, Nicolas Mansard†, Franck Multon‡, Julien Pettré§, and Dinesh Manocha*

UNC Chapel Hill* LAAS CNRS† Université Rennes II‡ INRIA§
http://gamma.cs.unc.edu/ITOMP

Abstract

We present an interactive motion planning algorithm to compute plausible trajectories for high-DOF human-like characters. Given a discrete sequence of contact configurations, we use a three-phase optimization approach to ensure that the resulting trajectory is collision-free, smooth, and satisfies dynamic balancing constraints. Our approach can directly compute dynamically balanced and natural-looking motions at interactive frame rates and is considerably faster than prior methods. We highlight its performance on complex human motion benchmarks corresponding to walking, climbing, crawling, and crouching, where the discrete configurations are generated from a kinematic planner or extracted from motion capture datasets.

Keywords: motion planning, dynamic balance, plausible motion

Concepts: •**Computing methodologies** → **Simulation by animation;** *Physical simulation; Motion processing;* Motion capture;

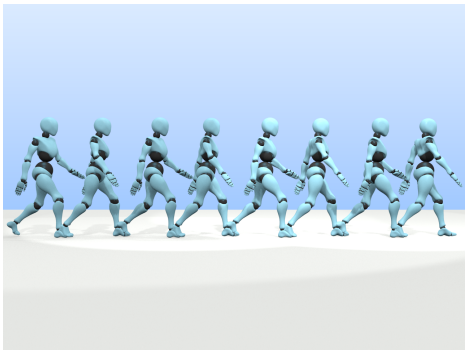


Figure 1: *We can compute the dynamically-balanced and natural looking trajectories for this* 58 *DOF human-model at interactive rates. The discrete end poses were generated using a Mocap dataset and we compute a continuous trajectory using our high DOF optimization algorithm.*

1 Introduction

Automatically synthesizing plausible motion animations for human like characters is one of the major challenges in computer graphics, in fields such as computer games, virtual reality, and computer animation. Furthermore, this problem is also important in robotics for humanoid motion planning and biomechanics. Some of the widely used approaches to generate natural looking motion use pre-recorded recorded data such as motion capture (MoCap) clips [Arikan and Forsyth 2002; Kovar et al. 2002; Treuille et al. 2007; Seol et al. 2013]. The MoCap database stores a set of motion data, which includes trajectories of joints or captured sensor locations based on the body. Each motion data set corresponds to an individual human who performs different actions (e.g. walking, running). The motions of the same action may also differ for different individuals.

One of the challenges is generating smooth and natural looking motions for different human actions. This involves using some discrete human poses in the MoCap data and generating continuous motion. A key issue is to ensure that the resulting motion or trajectory is dynamically stable or balanced and is natural-looking. Some of the simplest algorithms are based on procedural methods, but the resulting motion may not look natural [Geijtenbeek and Pronost 2012; Al Borno et al. 2014; Hämäläinen et al. 2015]. Other algorithms are based on the use of motion planning algorithms [Yamane et al. 2004]. In many cases, additional constraints include dealing with cluttered environments or maintaining contacts with the environment, e.g. a climbing wall or the human motion within a car.

This problem of generating dynamically balanced trajectories has also been studied in robotics and many solutions have been proposed based on optimization-based planning [Mordatch et al. 2012; Al Borno et al. 2013; Park and Manocha 2014; Wampler et al. 2014] or tree-based search [Bouyarmane and Kheddar 2011; Escande et al. 2013]. However, the complexity and running time of such algorithms can be high, especially as we consider multiple constraints, and resulting motions may not look plausible or are fast enough for interactive applications.

Instead of computing solutions that can simultaneously satisfy all of the constraints using a single framework, some approaches in robotics find the solutions using multiple steps, each taking into account different constraints. Our work is motivated by one such approach, *contacts-before-motion planning* [Kuffner et al. 2002; Escande et al. 2013]. The underlying planning algorithms first compute a feasible sequence of discrete contact configurations or stances from the initial position to the goal position. In the second step, a continuous trajectory is computed that interpolates those contact configurations and satisfies other constraints. In this paper, we mainly address the second step of continuous trajectory computation that simultaneously satisfies dynamic stability and natural-looking constraints [Khatib et al. 2004], in addition to generating collision-free motion in cluttered scenes.

Main Results: We present a novel trajectory optimization algorithm that uses decoupled planning to compute continuous trajectories from discrete contact configurations which are collision-free, dynamically balanced, and natural-looking. Our approach is

*e-mail:{chpark, jaesungp, dm}@cs.unc.edu
†e-mail:{stonneau, nmansard}@laas.fr
‡e-mail:{fmulton}@irisa.fr
§e-mail:{julien.pettre}@inria.fr

I3D '16, February 27-28, 2016, Redmond, WA
ISBN: 978-1-4503-4043-4/16/03
DOI: http://dx.doi.org/10.1145/2856400.2856405

based on high-dimensional trajectory optimization techniques that take into account many constraints. In order to handle the high-dimension of the configuration space and various constraints, our solution proceeds in three phases. We compute an initial trajectory from the discrete configurations that tends to minimize jerk motion. Secondly, we compute a collision-free trajectory using a parallel optimization algorithm with reduced DOFs. Finally, we perform a full DOF optimization that takes into account balancing and plausibility constraints. The dynamic balancing constraints are based on the equilibrium of forces and torques acting on the character body and can handle uneven terrains. We use energy-based formulation to generate plausible motion that minimizes jerk and erratic motion. We also parallelize the optimization algorithm on multiple cores to achieve interactive performance.

This three-phase algorithm enables us to compute continuous trajectories for high DOF human-like characters in complex environments corresponding to these human motions: climbing, crawling, and crouching. We have applied our approach to discrete configurations computed by a kinematic planner [Tonneau et al. 2015] and to poses extracted from motion capture datasets [Kovar et al. 2002] to compute motions with different interactions, including holding or pushing objects. As compared to prior methods, our algorithm offers two benefits:

- Our formulation is efficient and can perform trajectory planning on human-like characters with 34 or 58 DOF at interactive frame rates. It is significantly faster than prior methods that can handle contacts and balancing constraints using integrated approach.
- Our algorithm can directly compute dynamically balanced motion. In contrast, prior methods first compute a kinematic-stable trajectory, and refine it to compute a balanced trajectory in terms of quasi static balance.

Overall, our approach can be regarded as a hybrid combination of physics-based and data-driven approaches, which can be applied to Mocap data or the output of path planner. It can maintain the natural-looking characteristic and the style of the initial motion, while adapting to new characters or environments.

The rest of the paper is organized as follows. In Sec. 2, we give a brief survey of prior work. We present an overview of our trajectory planning algorithm in Sec. 3. We describe the details of the trajectory optimization in Sec. 4. Finally, we demonstrate our algorithm's performance in different application scenarios in Sec. 5 and analyse the results in Sec. 6.

2 Related Work

In this section, we give a brief overview of prior work in motion trajectory computation for dynamically balanced and plausible motions.

2.1 Physics-Based Motion Simulation

Many approaches use physics-based simulation techniques to synthesize physically plausible motion for different tasks [Wooten and Hodgins 2000; Seipel and Holmes 2005]. Such approaches can compute the motion trajectories for locomotion tasks such as walking or running using simplified physics constraints [Kajita et al. 2001; Kuo et al. 2005]. However, those simplified constraints (e.g. inverted pendulum for cyclic walking) are usually not applicable to new tasks, or require expertized new task specification.

For the general full-body motion computation, the balancing constraints are typically formulated using the equations of motion, which maintain the equilibrium among the internal and external forces exerted on the character: inertia, Coriolis, gravity, and contact-reaction forces. The balancing constraints are handled in the trajectory optimization [Mordatch et al. 2012; Posa and Tedrake 2013] or sampling-based search [Hämäläinen et al. 2015].

There is active research in other fields including robotics, biomechanics and neuroscience on characterizing plausible human motion [Mordatch et al. 2013]. In many ways, the notion of defining a plausible motion is subject to discussion. Our work is motivated by prior approaches that use energy efficient formulations for the motion synthesis [Tan et al. 2011]. In the remainder of this paper, we will refer to plausible motions as motions minimizing the energy criteria.

2.2 Data-Driven Motion Synthesis

There has been a lot of work to generate natural-looking motions [Kovar et al. 2002; Safonova et al. 2004; Ren et al. 2005]. Most of these methods are data-driven, which use Mocap (motion capture) human data to generate plausible motions. The idea of using pre-generated motions (i.e. with Mocap) has also been used for motion planning of humanoid robots, including predefined motion primitives [Liu et al. 2015] or blending of motion capture data [Pan et al. 2010]. Such approaches can compute natural-looking or stylized motions with relatively low computational costs [Ma et al. 2010], but have limitations in adapting to the new characters or environments that do not exist in the motion database.

2.3 Trajectory Optimization

Many motion computation techniques based on trajectory optimization have been proposed in the literature [Geijtenbeek and Pronost 2012]. Witkin and Kass [1988] proposed a space-time optimization framework to compute motion trajectories with user defined constraints and objectives. Some approaches directly integrate balancing constraints with contact planning into the optimization constraints. [Mordatch et al. 2012]. However, directly solving the optimization problem with both balancing and contact constraints can be challenging due to the high-dimensionality and many local minima, and therefore can take a long time or cannot find a feasible solution. In order to alleviate the problem, some approaches use Mocap data to constrain the search space [Liu et al. 2005].

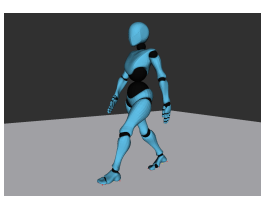
For applications that require real-time or interactive performance, techniques based on decoupled planners have been used. In the applications where the motion and the contacts are restricted to planar ground, the zero moment point (ZMP) [Dalibard et al. 2013] can be used to efficiently satisfy the balancing constraint. These methods compute a ZMP trajectory on the ground, and generate the motion of the entire character based on ZMP constraints. Another commonly used strategy is to first compute a kinematic-stable trajectory and refine it into a dynamically feasible trajectory [Kuffner et al. 2002]. However, these approaches tend to be more constrained and may not work well in complex scenarios.

3 Trajectory Planning

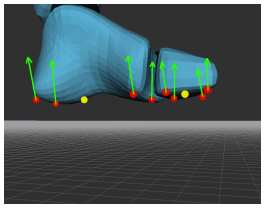
In this section, we first introduce the notation used in the paper and give an overview of our approach.

3.1 Notation and Assumptions

Although our approach can be used for any high-DOF and articulated characters, our goal is to compute trajectories for high DOF human-like characters that are dynamically balanced and natural



(a) *A high-DOF human-like character used in our benchmarks.*



(b) *Contact points (red) and contact forces (green) on a foot.*

Figure 2: *The character makes contact with the environment using the two feet. Contact forces* $\mathbf{f}_j$, ..., $\mathbf{f}_J$ *are optimized to ensure that the forces acting on the body are in equilibrium.*

looking as shown in Fig. 2(a). These high-DOF characters are represented using kinematic chains between the root and multiple free end-effectors.

We denote a character configuration of a single pose as a vector $\mathbf{q}$, which consists of the 6-DOF root and joint values. We define a motion trajectory, which is a function of time, as $\mathbf{q}(t)$. We represent $\mathbf{q}(t)$ using a matrix $\mathbf{Q} = [\mathbf{q}(t_1)\mathbf{q}(t_1), ..., \mathbf{q}(t_l)]$, which corresponds to the l configurations at the discretized keyframes with a fixed interval Δ_k. Many Mocap data formats are based on this configuration matrix representation, with an interval that is small enough to represent the originally continuous motion (e.g. $\Delta_k < 1/24$s). Similarly, the corresponding velocities are represented using a matrix $\dot{\mathbf{Q}}$. The trajectory value $\mathbf{q}(t)$ for $t_i < t < t_{i+1}$ is evaluated using a cubic interpolation function from $\mathbf{q}(t_i)$, $\dot{\mathbf{q}}(t_i)$, $\mathbf{q}(t_{i+1})$, and $\dot{\mathbf{q}}(t_{i+1})$. The input of our planner is a sequence of discrete contact configurations $\mathbf{p}_1, ..., \mathbf{p}_n$. Each input configuration is assumed to be feasible, to be balanced, and to have valid contacts with the objects in the environment (e.g. the ground).

We assume the character used in the planning has predefined end-effectors, e_i, which can make contacts with the environment (e.g. feet and hands for human-like characters). We assume that the contacts with the environment objects occur only with these end-effectors. The contact region of e_i is assumed to be planar, and represented by contact points c_i^j, which can exert contact forces $\mathbf{f}_i^j$ on the character (as shown by dots in red in Fig. 2(b)). We define an end-effector e_i is in-contact or has an active contact if one of contact forces $\mathbf{f}_i^j$ is non-zero.

Our goal is to compute a continuous trajectory $\mathbf{Q}^*(t)$ for the human-like character that is collision-free, dynamically balanced, and plausible. At the same time, it satisfies constraints corresponding to the joint positions, velocity, and acceleration limits (i.e. kinematic and dynamic constraints) of the character.

3.2 Multi-Step Trajectory Optimization

The input configuration sequences can be generated from MoCap data or many known path planning algorithms. The specific algorithm used in our work will be described in Sec. 5. Given the n input configurations, $\mathbf{p}_1, ..., \mathbf{p}_n$, we generate an initial trajectory $\mathbf{Q}$ that is used by our multi-stage optimization algorithm.

In the n input configurations, we also compute whether each contact point is in-contact. In order to compute the stability constraints, we compute the initial contact forces of the active contacts that satisfy the static stability constraint, and interpolate the force values along the trajectory.

Although the n input configurations, generated using path planning or captured using Mocap data, may be feasible in terms of collision-free, stability and plausibility constraints, the interpolated

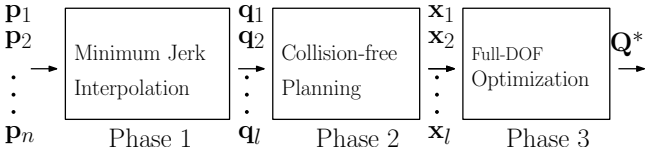


Figure 3: *We highlight different components of our multi-step trajectory optimization planner. The input configurations* $\mathbf{p}_1$, ..., $\mathbf{p}_n$ *are interpolated to generate the initial trajectory* $\mathbf{Q} = \mathbf{q}_1, ..., \mathbf{q}_l$. *A parallel optimization is used to compute a collision-free trajectory, and the subsequent step optimizes it with all constraints, including balancing and plausible motion.*

additional $m(n-1)$ configurations may violate some of these constraints. As a result, we use a trajectory optimization approach to compute a feasible trajectory. Furthermore, as the constraints have different characteristics, we optimize the trajectory in multiple stages, where each phase deals with different constraints.

In the first optimization phase, we compute a collision-free continuous trajectory from the initial trajectory if it is not collision-free. Since the collision-free space is discontinuous and has many local minima, a global search in the configuration space is required to compute a collision-free solution. We use a parallel optimization algorithm with different random seeds that tries to compute multiple trajectories to avoid being stuck in local optima. In order to constrain the search space, we only optimize the joint values on the kinematic chains between the root of the model representation and free end-effectors that do not have an active contact. This phase allows quick computation of the collision-free trajectory that is close to the initial trajectory.

In the next optimization phase, we optimize the joint values along with contact forces of the entire trajectory by taking into account dynamic balance and plausibility constraints. Our dynamic stability constraints are based on the equilibrium of the wrenches, forces and torques, acting on the character. We compute the sum of the external forces, including gravity force, reaction force, etc., and the internal forces, including joint forces, inertial force, etc., and the resulting torques that are exerted on the character. The plausibility constraints tend to minimize the joint torques to prevent jerky or unnecessary motions. These constraints have coupling effects among them (such as minimizing torques and adding contact forces for the balancing affect each other), but a feasible solution is computed using local optimization. The details of the constraints are discussed in Sec. 4.

4 Multi-Stage Trajectory Optimization

In this section, we present the details of our planning algorithm that computes feasible trajectories from discrete configurations. We first present the generation of the initial trajectory from the input configurations. Next, we describe a multiple-phase optimization to compute collision-free, balanced and natural-looking trajectories.

4.1 Initial Trajectory Generation

Our planner uses a sequence of discrete configurations $\mathbf{p}_1, ..., \mathbf{p}_n$ as an input. Given this input, we generate a trajectory that is used as an input for multi-phase optimization. The initial trajectory $\mathbf{Q}$ is represented using keyframe configurations $\mathbf{q}_1, ..., \mathbf{q}_l$, where $l = (nm + n - m)$ as we add m configurations between each pair of adjacent input configurations. We initialize $\dot{\mathbf{q}}_i = \mathbf{0}$ for the keyframes that correspond to one of the input configurations. The internal keyframes are computed using a minimum jerk trajectory computation algorithm, which is a commonly used model

in neuromuscular studies for smooth human trajectories [Flash and Hogan 1985]. Each interpolated trajectory corresponds for $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ is initialized using the quintic curve $\{\mathbf{f}(t)|0 \leq t \leq 1\}$, where $\mathbf{f}(0) = \mathbf{p}_i, \mathbf{f}(1) = \mathbf{p}_{i+1}$, and $\dot{\mathbf{f}}(0) = \dot{\mathbf{f}}(1) = \ddot{\mathbf{f}}(0) = \ddot{\mathbf{f}}(1) = 0$.

4.2 Collision-Free Planning of Reduced-DOF Motions

Although the input configurations are collision-free, the interpolated trajectory $\mathbf{Q}$ may have collisions with the objects in the environment or self-collisions between different links. In the first planning step, we compute a collision-free trajectory using trajectory optimization. Instead of optimizing the entire DOF of the character, we compute solutions using a reduced DOF for efficiency reasons. For a trajectory segment $[\mathbf{q}_{kj}, ..., \mathbf{q}_{(k+1)j}]$ that is generated by the interpolation of $\mathbf{p}_j$ and $\mathbf{p}_{(j+1)}$, some end-effectors may have active contacts that exert contact forces on the human-like figure, and their positions are not changed in $[\mathbf{q}_{kj}, ..., \mathbf{q}_{(k+1)j}]$. Most collisions tend to occur for floating end-effectors that do not have active contacts. Therefore, we only optimize the joint positions that lie on the kinematic chains of end-effectors that have inactive contacts. We formulate the objective function for this computation as

$$\mathbf{Q}^* = \arg\min_{\mathbf{Q}} \sum_t (C_{col}(\mathbf{q}_t) + w\|\ddot{\mathbf{q}}_t\|^2), \tag{1}$$

where $\mathbf{q}_t$ corresponds to the interpolated configurations from trajectory keyframes $\mathbf{Q}$ and w is the weight for the trajectory smoothness cost that is used to avoid jerky motions.

$C_{col}(\mathbf{q}_t)$ corresponds to the constraint cost function for collisions in our optimization formulation. It is computed using the mesh objects $BV(\mathbf{q}_t)$ that represent the character body, where $BV(\mathbf{q}_t)$ corresponds to the BV located at configuration $\mathbf{q}_t$, and the set $\mathcal{W}$ of the mesh objects in the environment. $C_{col}(\mathbf{q}_t)$ has to be $\mathbf{0}$ in order to avoid collisions with the objects in the environment or self-collisions for the character. We formulate this constraint as:

$$\begin{aligned} C_{col}(\mathbf{q}_t) = &\sum_{\substack{BV \in A \\ E \in \mathcal{W}}} PD(BV(\mathbf{q}_t), E)^2 \\ &+ \sum_{\substack{BV_i, BV_j \in A \\ BV_i \neq BV_j}} PD(BV_i(\mathbf{q}_t), BV_j(\mathbf{q}_t))^2, \end{aligned} \tag{2}$$

where $PD(O_1, O_2)$ is the penetration depth between mesh objects O_1 and O_2, which refers to the extent of inter-penetration between two overlapping objects. If the objects don't overlap or just touch, their penetration depth is zero.

In order to compute collision-free solutions using the constraint function $C_{col}(\mathbf{q}_t)$, simple local optimization methods may not work well and can get stuck in local minima. Instead, we use a parallel stochastic optimization algorithm that computes multiple trajectories [Park et al. 2012] to improve the performance and the probability of finding the global minima.

4.3 Full-DOF Optimization with Constraints

In the next step of our planner, we use constrained optimization to optimize the trajectory $\mathbf{Q}$ with cost functions that correspond to all constraints required for the feasible solution. These include constraints corresponding to collision-free motion, dynamic stability, and plausibility.

While the optimization for collision-free planning only requires the configuration $\mathbf{q}$, the optimization of our second phase requires additional parameters, $\mathbf{f}_i$, which corresponds to the contact forces, as defined in Sec. 3.1. With these parameters, the optimization variables for a keyframe i are defined as follows:

$$\mathbf{x}_i = \left[\mathbf{q}_i, \dot{\mathbf{q}}_i, \mathbf{f}_i, \dot{\mathbf{f}}_i\right]. \tag{3}$$

The objective function of this optimization is given as

$$\min_{\mathbf{x}_1, ..., \mathbf{x}_l} \sum_t C(\mathbf{x}_t), \tag{4}$$

where $C(\mathbf{x}_i)$ represents the cost function for the parameter $\mathbf{x}_i$ at time t. This cost function is decomposed as:

$$f(\mathbf{x}_i) = C_{col}(\mathbf{q}_i) + C_{ds}(\mathbf{x}_i) + C_{pla}(\mathbf{x}_i), \tag{5}$$

where $C_{col}(\mathbf{q}_i)$, $C_{ds}(\mathbf{x}_i)$, $C_{pla}(\mathbf{x}_i)$ represent the costs for the collision-free constraint, dynamic stability constraint, and plausible motion constraint, respectively.

4.3.1 Collision-free Constraint

We use the same collision constraint function $C_{col}(\mathbf{q}_i)$ defined above. The goal is to ensure that the configurations generated during this optimization phase are still collision-free.

4.3.2 Dynamically Balanced Constraint

The stability constraints can be evaluated using the Newton-Euler equation [Trinkle et al. 1997], based on external forces (gravity force, reaction force, etc.) and internal forces (joint forces, inertial force, etc.) that are exerted on the body. A pose is dynamically balanced if all the exerting forces and torques result in an equilibrium. The reaction forces can exist only if the corresponding end-effector makes an active contact. As shown in Fig.2(b), an end-effector contact is approximated by multiple contacts in a plane, and each contact exerts its own contact force. Therefore the stability cost function $C_{ds}(\mathbf{x}_i)$ is formulated as follows:

$$C_{ds}(\mathbf{x}_i) = \| \sum^{J} w_c(\mathbf{x}_i) + w_g(\mathbf{q}_i) + w_i(\mathbf{q}_i) \|^2, \tag{6}$$

where J is the total number of contact points, $w_c(\mathbf{x}_i)$ is the contact wrench for $\mathbf{x}_i$, and $w_g(\mathbf{q}_i)$ and $w_i(\mathbf{q}_i)$ correspond to the gravity and inertia wrenches for a configuration $\mathbf{q}_i$, respectively. $w_c(\mathbf{x}_i)$ is set to 0 for non-active contacts.

4.3.3 Plausible Motion Constraint

As we discussed in Sec. 2.1, we use the torque minimization [Lo et al. 2002] constraint to compute the plausible motions. In particular, we use the inverse dynamics to compute the joint torque for the configuration $\mathbf{q}_t$ and the contact forces $\mathbf{f}_t$, and formulate the constraint cost as the squared sum:

$$C_{pla}(\mathbf{x}_i) = \sum_j w(j) \cdot \|\tau_j(\mathbf{x}_t)\|^2, \tag{7}$$

where $\tau_j(\mathbf{x}_t)$ is the joint torque of the j-th joint.

4.4 Parallelization

We solve the non-linear optimization problem corresponding to (4) using the L-BFGS [Nocedal 1980] algorithm, which is a quasi-Newton method with an approximated Hessian. We compute numerical derivatives with box constraints of the variables that correspond to the joint position and velocity limits. The optimization

problem in (4) has $l \times (\text{length}(\mathbf{q}_i) + \text{length}(\mathbf{f}_i)) \times 2$ parameters, where the lengths of $\mathbf{q}_i$ and $\mathbf{f}_i$ corresponds to the number of joints, and $3\times$ of the number of body contact points.

Because some of the constraints in the optimization formulation (e.g. collision-free constraints) do not have analytical derivatives, we approximate using numerical derivatives. Numerical derivative evaluation can be compute-intensive, as it requires $2\times$(number of optimization parameters) cost evaluations. We improve the runtime performance by: (1) using parallel computation using multi-core CPU threads; (2) reducing redundant computations; and (3) load balancing by rearranging the evaluation order.

As described in Sec. 3, our trajectory is represented as a set of piecewise cubic curves. Therefore, a change in a keyframe only impacts a local region with $(2m + 1)$ parameters. It reduces the required evaluations for the numerical derivatives. The optimization parameters consist of 4 different vectors, but not all parameters affect all the constraints. For example, the collision-free constraint only depends on the joint configuration $\mathbf{q}$. When we change other vectors, $\mathbf{v}$ and $\dot{\mathbf{v}}$, collision checking is not required. Note that $\dot{\mathbf{q}}$ only affects the poses for the interpolated frames. Furthermore, we also observe performance speed up from reordering the parameter evaluation. Since collision checking dominates the computation time in complex environments, we rearrange the order of parameter evaluation to ensure that the cost of collision evaluation is almost equal within each thread. As shown in Sec. 5, our parallel implementation achieves almost linear speedup in terms of the number of threads.

5 Results

In this section, we highlight the performance of our algorithm on different benchmarks. Our motion planner enables us to compute dynamically balanced and plausible motion for different characters. The accompanying video highlights the motions computed by our algorithm. In order to evaluate our approach, we used discrete input configurations generated using two different sources, a collision-free path planning algorithm and sampled poses from MoCap data. Table 1 presents the complexity of the benchmarks and the performance of our planning results.

5.1 Kinematic-Stable Pose Sequences

The input pose sequences for the first benchmark set are computed using a reachability-based PRM (RB-PRM) [Tonneau et al. 2015]. RB-PRM computes acyclic balanced discrete poses using a random sampling to search in a low dimensional subset of the entire configuration space, which is chosen such that the character is close enough to the environment and maintains a contact with the environment.

The human-like character used in the experiments has 34 DOFs, which are decomposed as 6 root DOFs and has 7 DOFs for each limb. We evaluate the performance of our planner on three benchmark scenarios, where the input discrete pose sequences are computed using RB-PRM. The complete trajectories generated by our algorithm are shown in the video.

Climbing Blocks: The input configuration is climbing on a wall using several blocks on the wall. We compute the trajectory with the collision, balancing, and plausibility constraints. Fig. 6(a) highlights the initial interpolated trajectory, the collision-free trajectory after the first optimization, the final trajectory with dynamic balancing and the plausible motion constraints, respectively.

Crawling on Obstacles: The character goes from a standing to a crouching position to pass under an obstacle (i.e. collision-free motion). The space between the obstacle and the ground is narrow, which makes it difficult to find a collision-free trajectory (see Fig.6(b)). Many prior methods would not work well in such environments.

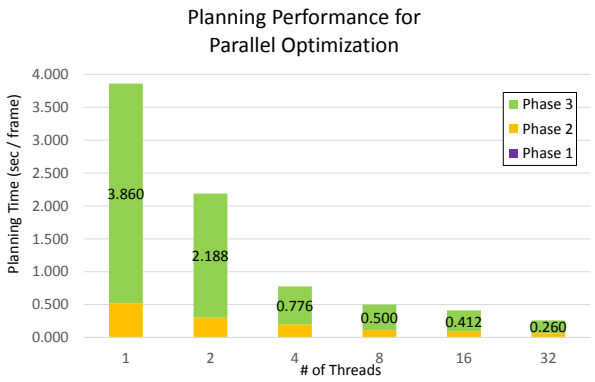


Figure 4: *The planning performance of our parallel optimization. We present the planning time per frame for* Climbing *benchmark with a different number of threads. The results show the planning time decreases near-linearly with the number of threads.*

Escaping from a Truck: The character crawls through the front window of a truck (see Fig.6(c)). We demonstrate the trajectory computed by our multi-phase optimization approach.

5.2 MoCap Datasets

Our second set of benchmarks use sampled pose sequences from MoCap data. We extract only two configurations for walking and pushing motions from motion capture data for the model, which have contacts with both feet. The computed walking and pushing motion for the human-model are shown in Fig. 7(a) and (b), which are similar to the original MoCap data. In order to validate the dynamic balancing and the plausibility constraints of our approach, we computed the continuous trajectories from the poses for the character with a different mass. As we add more mass to the right arm by adding a suitcase or a heavy iron beam (Fig. 7(c)), the walking motion lowers the right arm down more or produces bigger upperbody movements, respectively. The complete trajectories generated by our algorithm are shown in the video.

6 Analysis

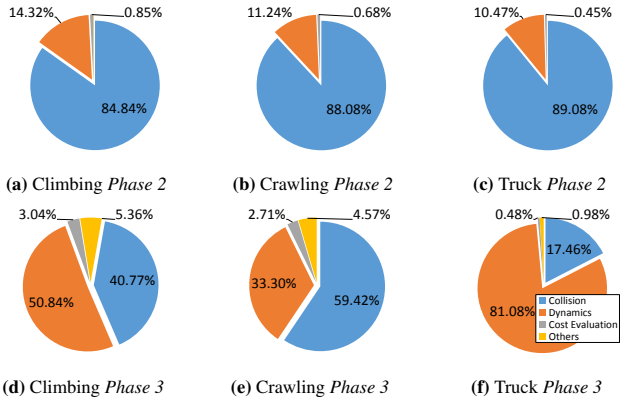
In this section, we first analyze the performance of the benchmark results, and discuss the limitation of the proposed approach. Next, we compare our approach with related approaches.

6.1 Scalability with Parallel Optimization

We execute our planner with different numbers of threads and highlight the benefits of parallelization in Fig. 4. We present the average trajectory planning time per frame for *Climbing* benchmark with different numbers of threads. The graph shows that the planning time decreases near-linearly with the number of threads. It also demonstrates the planning time for each phase. Phase 1 (initial trajectory generation) takes a relatively small fraction of the frame time. However, the computation time for phase 2 (collision-free planning) and phase 3 (trajectory optimization) are reduced, as the number of threads increases.

	Data Source	# of joints	# of discrete poses	# of frames	Average trajectory planning time / frame
Climbing Blocks	RB-PRM	34	16	481	0.308 sec
Escaping from a Truck	RB-PRM	34	12	353	.289 sec
Crawling on Obstacles	RB-PRM	34	20	609	0.459 sec
Walking	MoCap Data	58	2	64	0.413 sec
Iron Beam	MoCap Data	58	2	64	0.532 sec
Pushing	MoCap Data	58	2	64	0.471 sec

Table 1: *Model complexity and the performance of trajectory planning: We highlight the complexity of each benchmark in terms of number of joints, the number of input discrete poses, and the number of frames that is governed by the length of the motion. We compute the average trajectory planning time per frame for each benchmark on a multi-core PC.*



(a) Climbing *Phase 2* **(b)** Crawling *Phase 2* **(c)** Truck *Phase 2*

(d) Climbing *Phase 3* **(e)** Crawling *Phase 3* **(f)** Truck *Phase 3*

Figure 5: *The timing breakdown between different components for our benchmarks.*

6.2 Performance Analysis

In Fig. 5, we highlight the timing breakdown of the total trajectory planning time for three benchmarks. The result demonstrates that collision-checking dominates the collision-free planning within phase 2. Phase 3 computes a dynamically balanced and plausible trajectory, while maintaining the collision-free constraint. In phase 3, only in *Crawling* benchmark, the collision checking takes more than half of the total time. *Crawling* benchmark has a narrow passage, which makes collision-free path finding harder. However, *Climbing* and *Truck* benchmarks have more free or open space for the character to move its limbs to avoid collisions.

6.3 Limitations

Our motion optimization algorithm has a few limitations. We choose the jerk minimization criteria in our approach, as described in Sec. 4.1. This criteria sets the velocities and accelerations of the keyframes to be zeros for the initial trajectory computation. This formulation works well for poses where the limbs are in contact with obstacles during the keyframes (e.g. climbing blocks benchmark). For other cases, e.g. the arm positions in the MoCap-based walking benchmarks, the floating end-effectors of the limbs with zero velocities and accelerations may not appear natural. As a result, we need better formulation of constraints that can result in plausible motion, including (Eq.(7)) that is formulated as torque minimization.

In Section 3.1, we assume that contact regions between the end-effectors and the obstacles are planar, which allows efficient contact force computation. However, some of the input poses used in our benchmarks have non-planar contacts. The computed poses from the RB-PRM specify the contacts between the limbs and the objects as single points, and the poses from MoCap data have non-planar contact surfaces, which can arise from deformation of hands and feet. We use inverse kinematics (IK) to preprocess the active contacts from input poses so that the resulting contacts with the obstales are planar, but this may violate the balancing constraint.

The trajectory optimization algorithm uses box constraints for the joint position and velocity limits, but the maximum torques of the joints are not constrained in the computed trajectory. Therefore, the computed motions make no assumption about the strength of human arms. For example, in the Walking benchmark with suitcases, the right arm is lowered till the suitcase does not collide with the right leg, but still maintains the balance. Adding torque limits can improve the naturalness of the motions with a trade-off in the planning performance.

6.4 Comparisons with Related Approaches

Computing human-like motion using trajectory optimization can be time consuming, even with relaxed dynamics constraints [Mordatch et al. 2012]. Data-driven motion synthesis approaches use precomputation of MoCap data to compute physics-based motion [Liu et al. 2005] or a variety of motions [Ma et al. 2010] that have the same style as the input motion. This can take long computation time or has limited motion applicability (e.g. only applicable for walking or running motions). Furthermore, these approaches only consider ground contacts, and collision-free geometric constraints, which can be expensive to compute, are not taken into account.

MoCap-based humanoid robot planning methods [Pan et al. 2010; Liu et al. 2015] focus on computing collision-free and balanced motions for human-like robots or characters that tends to look natural. However, they use a manual setup of contact poses, which can be limiting. In contrast to these approaches, our approach can compute plausible human-like motions that can adapt to e new characters or environments at interactive rates, and does not have a large precomputation overhead.

7 Conclusions and Future Work

We present a novel motion planning algorithm that computes dynamically balanced and stable trajectories for discrete contact configurations. We use a multi-phased optimization strategy based on a decoupled planner and are able to compute collision-free trajectories that satisfy all these constraints. As compared to prior methods, our approach is much faster and the resulting motion trajectories are dynamically balanced as opposed to quasi static motions. We highlight the performance on complex human motion benchmarks corresponding to walking, climbing, crawling, and crouching. Furthermore, we parallelize the performance to obtain interactive performance.

As we discussed in Sec. 6.3, our approach has some limitations

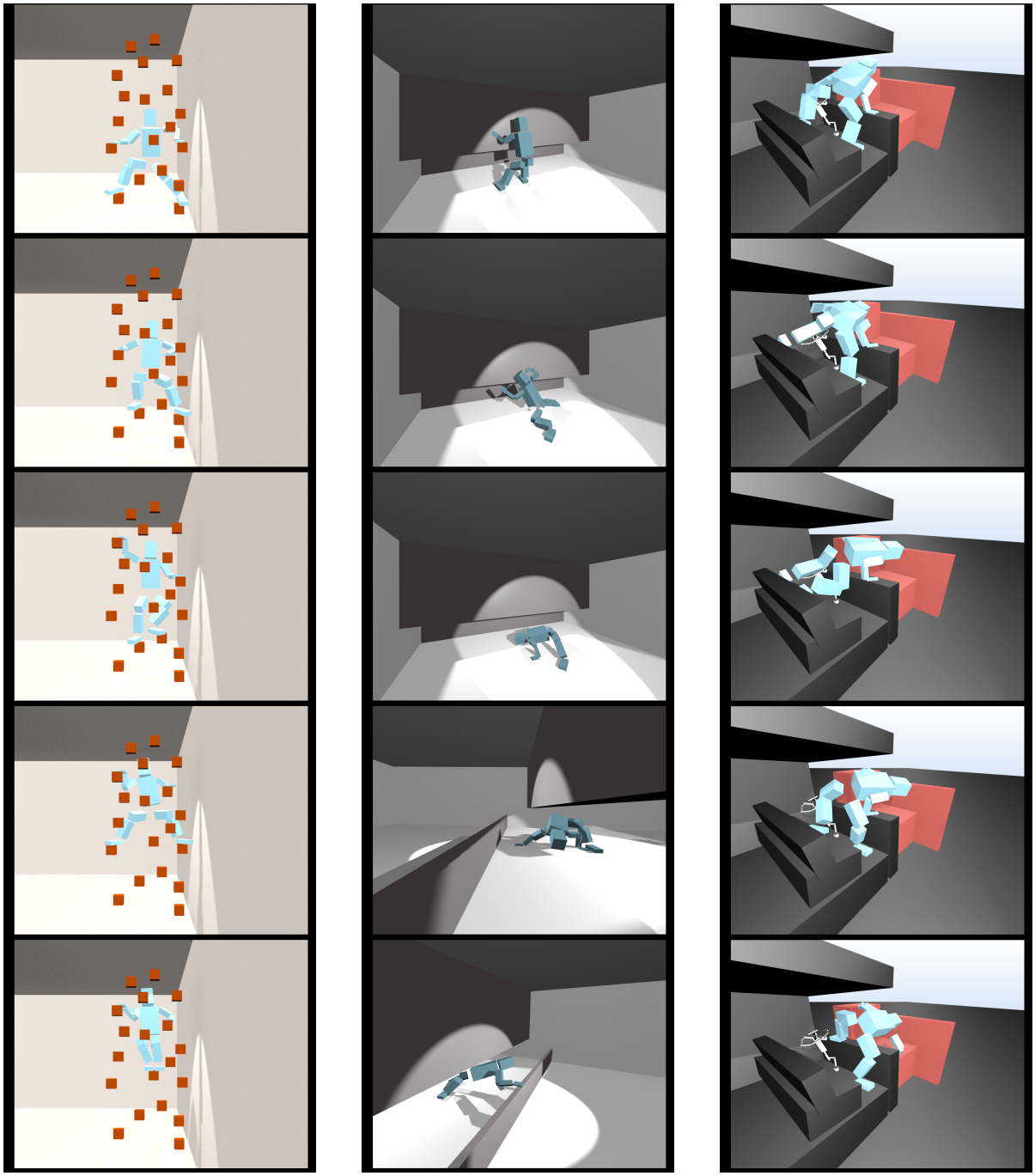


Figure 6: *The computed trajectories for the (a) Climbing, (b) Crawling and (c) Truck benchmarks.*

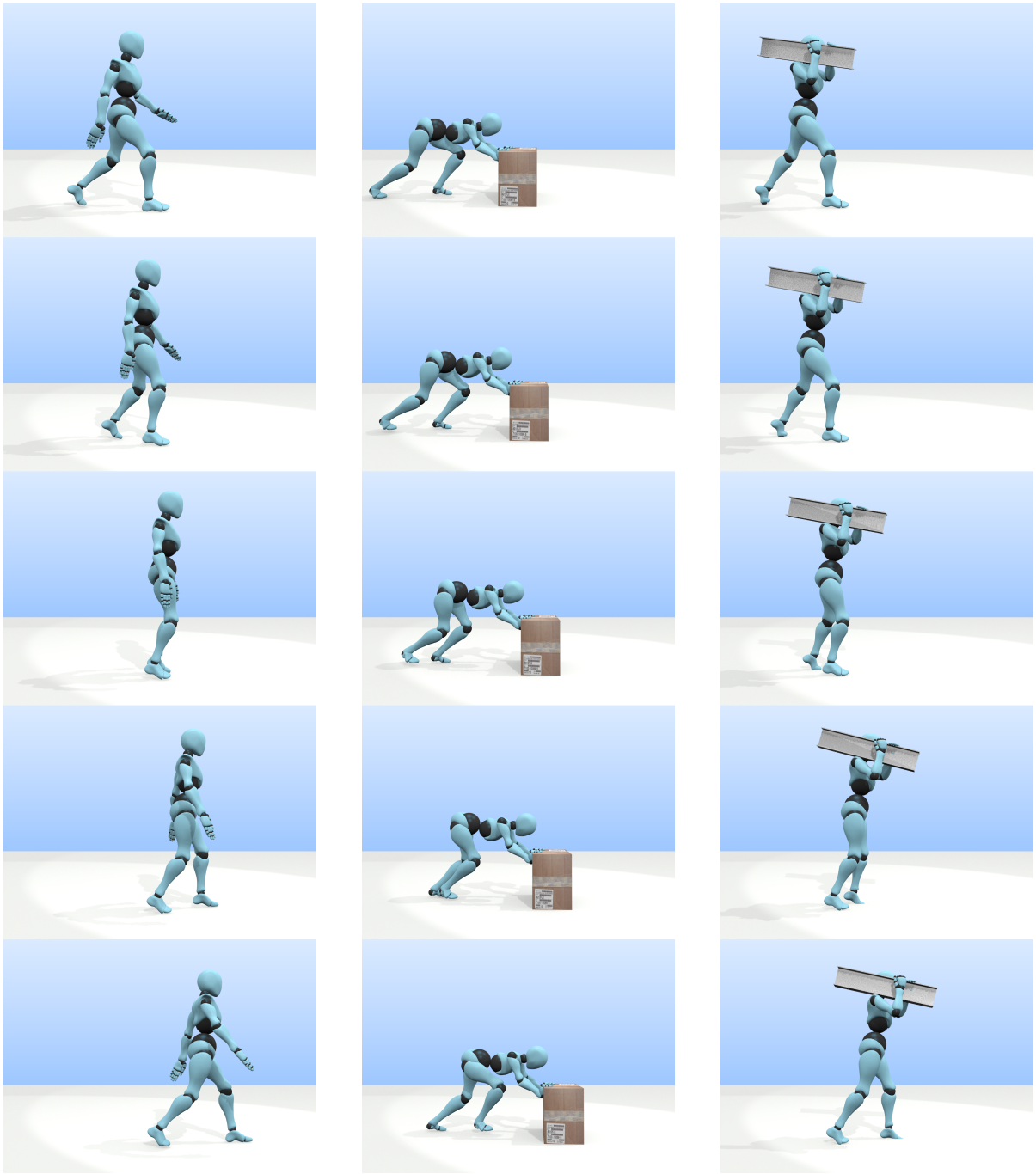


Figure 7: *The computed trajectories for the (a) Walking, (b) Pushing and (c) Holding benchmarks.*

in terms of the naturalness of the computed motions. While the decoupled planner approach is faster, it searches for the solution in a more constrained space. The use of parallel trajectory optimization increases the chances of a global solution, but it can still get stuck in local minima. In terms of future work, we would like to overcome these limitations. We would also like to evaluate the performance of our algorithm for other human motions and actions.

8 Acknowledgments

We are grateful for the feedback provided by David Kasik, Christopher Senesac and Timothy Sikora at Boeing. This research is supported in part by ARO Contract W911NF-14-1-0437, NSF award 1305286, a grant from Boeing, Euroc (project under FP7 Grant Agreement 608849) and Entracte (ANR grant agreement 13-CORD-002-01).

References

AL BORNO, M., DE LASA, M., AND HERTZMANN, A. 2013. Trajectory optimization for full-body movements with complex contacts. *Visualization and Computer Graphics, IEEE Transactions on 19*, 8, 1405–1414.

AL BORNO, M., FIUME, E., HERTZMANN, A., AND DE LASA, M. 2014. Feedback control for rotational movements in feature space. In *Computer Graphics Forum*, vol. 33, Wiley Online Library, 225–233.

ARIKAN, O., AND FORSYTH, D. A. 2002. Interactive motion generation from examples. In *ACM Transactions on Graphics (TOG)*, vol. 21, ACM, 483–490.

BOUYARMANE, K., AND KHEDDAR, A. 2011. Multi-contact stances planning for multiple agents. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, 5246–5253.

DALIBARD, S., EL KHOURY, A., LAMIRAUX, F., NAKHAEI, A., TAIX, M., AND LAUMOND, J.-P. 2013. Dynamic walking and whole-body motion planning for humanoid robots: an integrated approach. *The International Journal of Robotics Research*.

ESCANDE, A., KHEDDAR, A., AND MIOSSEC, S. 2013. Planning contact points for humanoid robots. *Robotics and Autonomous Systems 61*, 5, 428–442.

FLASH, T., AND HOGAN, N. 1985. The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience 5*, 7, 1688–1703.

GEIJTENBEEK, T., AND PRONOST, N. 2012. Interactive character animation using simulated physics: A state-of-the-art review. In *Computer Graphics Forum*, vol. 31, Wiley Online Library, 2492–2515.

HÄMÄLÄINEN, P., RAJAMÄKI, J., AND LIU, C. K. 2015. Online control of simulated humanoids using particle belief propagation. In *Proc. SIGGRAPH '15*, ACM, New York, NY, USA.

KAJITA, S., MATSUMOTO, O., AND SAIGO, M. 2001. Real-time 3d walking pattern generation for a biped robot with telescopic legs. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 3, IEEE, 2299–2306.

KHATIB, O., WARREN, J., DE SAPIO, V., AND SENTIS, L. 2004. Human-like motion from physiologically-based potential energies. In *On advances in robot kinematics*. Springer, 145–154.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. In *ACM transactions on graphics (TOG)*, vol. 21, ACM, 473–482.

KUFFNER, J. J., KAGAMI, S., NISHIWAKI, K., INABA, M., AND INOUE, H. 2002. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots 12*, 1, 105–118.

KUO, A. D., DONELAN, J. M., AND RUINA, A. 2005. Energetic consequences of walking like an inverted pendulum: step-to-step transitions. *Exercise and sport sciences reviews 33*, 2, 88–97.

LIU, C. K., HERTZMANN, A., AND POPOVIĆ, Z. 2005. Learning physics-based motion style with nonlinear inverse optimization. In *ACM Transactions on Graphics (TOG)*, vol. 24, ACM, 1071–1081.

LIU, C., ATKESON, C. G., FENG, S., AND XINJILEFU, X. 2015. Full-body motion planning and control for the car egress task of the darpa robotics challenge. In *Humanoid Robots (Humanoids), 2015 15th IEEE-RAS International Conference on*.

LO, J., HUANG, G., AND METAXAS, D. 2002. Human motion planning based on recursive dynamics and optimal control techniques. *Multibody System Dynamics 8*, 4, 433–458.

MA, W., XIA, S., HODGINS, J. K., YANG, X., LI, C., AND WANG, Z. 2010. Modeling style and variation in human motion. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, 21–30.

MORDATCH, I., TODOROV, E., AND POPOVIĆ, Z. 2012. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG) 31*, 4, 43.

MORDATCH, I., WANG, J. M., TODOROV, E., AND KOLTUN, V. 2013. Animating human lower limbs using contact-invariant optimization. *ACM Transactions on Graphics (TOG) 32*, 6, 203.

NOCEDAL, J. 1980. Updating quasi-newton matrices with limited storage. *Mathematics of computation 35*, 151, 773–782.

PAN, J., ZHANG, L., LIN, M. C., AND MANOCHA, D. 2010. A hybrid approach for simulating human motion in constrained environments. *Computer Animation and Virtual Worlds 21*, 3-4, 137–149.

PARK, C., AND MANOCHA, D. 2014. Fast and dynamically stable optimization-based planning for high-dof human-like robots. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, IEEE, 309–315.

PARK, C., PAN, J., AND MANOCHA, D. 2012. ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In *Proceedings of International Conference on Automated Planning and Scheduling*.

POSA, M., AND TEDRAKE, R. 2013. Direct trajectory optimization of rigid body dynamical systems through contact. In *Algorithmic Foundations of Robotics X*. Springer, 527–542.

REN, L., PATRICK, A., EFROS, A. A., HODGINS, J. K., AND REHG, J. M. 2005. A data-driven approach to quantifying natural human motion. *ACM Transactions on Graphics (TOG) 24*, 3, 1090–1097.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM Transactions on Graphics (TOG)*, vol. 23, ACM, 514–521.

SEIPEL, J. E., AND HOLMES, P. 2005. Running in three dimensions: Analysis of a point-mass sprung-leg model. *The International Journal of Robotics Research 24*, 8, 657–674.

SEOL, Y., O'SULLIVAN, C., AND LEE, J. 2013. Creature features: online motion puppetry for non-human characters. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, 213–221.

TAN, J., GU, Y., TURK, G., AND LIU, C. K. 2011. Articulated swimming creatures. In *ACM Transactions on Graphics (TOG)*, vol. 30, ACM, 58.

TONNEAU, S., MANSARD, N., PARK, C., MANOCHA, D., MULTON, F., AND PETTRÉ, J. 2015. A reachability-based planner for sequences of acyclic contacts in cluttered environments. In *Proceedings of International Symposium on Robotics Research*.

TREUILLE, A., LEE, Y., AND POPOVIĆ, Z. 2007. Near-optimal character animation with continuous control. In *ACM Transactions on Graphics (TOG)*, vol. 26, ACM, 7.

TRINKLE, J. C., PANG, J.-S., SUDARSKY, S., AND LO, G. 1997. On dynamic multi-rigid-body contact problems with coulomb friction. *ZAMM-Journal of Applied Mathematics and Mechanics 77*, 4, 267–279.

WAMPLER, K., POPOVIĆ, Z., AND POPOVIĆ, J. 2014. Generalizing locomotion style to new animals with inverse optimal regression. *ACM Transactions on Graphics (TOG) 33*, 4, 49.

WITKIN, A., AND KASS, M. 1988. Spacetime constraints. In *ACM Siggraph Computer Graphics*, vol. 22, ACM, 159–168.

WOOTEN, W. L., AND HODGINS, J. K. 2000. Simulating leaping, tumbling, landing and balancing humans. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1, IEEE, 656–662.

YAMANE, K., KUFFNER, J., AND HODGINS, J. K. 2004. Synthesizing Animations of Human Manipulation Tasks. *ACM Trans. on Graphics (Proc. SIGGRAPH 2004)*.